REMARKS

The present amendment is submitted in response to the Office Action received from the United States Patent Office dated January 7, 2009. The Patent Office has rejected Claims 1-14 and 31-32 under 35 U.S.C. § 103(a) as being unpatentable over Advanced Compiler Design & Implementation, Steven S. *Muchnick*, August 19, 1997 in view of *Tip et al.* (U.S. Patent Number 7,003,507). Finally, the Patent Office has rejected Claims 15-24, 33, and 34 under 35 U.S.C. § 103(a) as being unpatentable over Fast Static Analysis of C++ Virtual Function Calls, David F. *Bacon et al.*, ACM, 1996, pages 324-341 view of *Tip et al.*

In response to the Office Action, Applicant has amended Claims 1, 8, 15 and 20. Applicant respectfully submits that the amendments to the claims and the explanations below overcome the rejections to the claims. Applicant submits that all of the claims are now in condition for allowance. Notice to that effect is requested.

The Patent Office rejected Claims 1-14 and 31-32 under 35 U.S.C. § 103(a) as being unpatentable over Advanced Compiler Design & Implementation, Steven S. *Muchnick*, (herein after *Muchnick*) August 19, 1997 in view of *Tip et al.* (U.S. Patent Number 7,003,507). The Patent Office states that as to Claim 1, *Muchnick* teaches a method for analyzing a program, comprising: determining a set of functions required by the program by performing local type constraint analysis at intermediate language instruction level (*Munchnick*, page 609-618, CFG). The Patent Office alleges that *Muchnick* does not explicitly teach that the analysis is performed to determine which functions have the potential of being executed and determining a call path that may reach a function containing such instructions. However, the Patent Office states that *Tip et al.* teaches determining the reachability of local methods so that unreachable methods can be removed (i.e. col. 1 lines 40-53). The Patent Office states that it would have been obvious for one having ordinary skill in the art to modify *Muchnick's* disclosed system to incorporate the teachings of *Tip*.

Amended Claim 1 requires a method for analyzing a program, comprising: providing an application builder that receives source code instructions and determines the minimum amount of code that is required by a program; determining a set of functions required by the program by performing local type constraint analysis at intermediate language instruction level to determine which functions have the potential of being executed; and determining a call path that may reach a function containing such instruction.

Amended claim 8 requires a computer-readable medium storing computer-executable process steps of a process for analyzing a program, comprising: determining a set of functions required by the program by performing local type constraint analysis at intermediate language instruction level to determine which functions have the potential of being executed and eliminating unused functions whereby the program is analyzed recursively to determine which functions are called throughout the program; and determining a call path and a value graph that may reach a function containing such instruction.

Neither *Advanced Compiler Design & Implementation* or *Tip et al.* teaches or suggest determining a set of functions required by the program by performing local type constraint analysis at intermediate language instruction level to determine which functions have the potential of being executed and determining a call path that may reach a function containing such instruction as required by Claim 1 and 8. Moreover, neither *Advanced* or *Tip et al.* teach an application builder that receives source code instructions and determines the minimum amount of code required by a program as required by Claim 1. Further, neither *Advanced* or *Tip et al.* teach or suggest eliminating unused functions whereby the program is analyzed recursively to determine which functions are called throughout the program as required by Claim 8.

It is further submitted that the question under §103 is whether the totality of the art would collectively suggest the claimed invention to one of ordinary skill in this art. *In re Simon*, 461 F.2d 1387, 174 USPQ 114 (CCPA 1972).

That elements, even distinguishing elements, are disclosed in the art is alone insufficient. It is common to find elements somewhere in the art. Moreover, most if not all elements perform their ordained and expected functions. The test is whether the invention as a whole, in light of the teaching of the reference, would have been obvious to one of ordinary skill in the art at the time the invention was made. *Connell v. Sears, Roebuck & Co.*, 722 F.2d 1542, 220 USPQ 193 (Fed. Cir. 1983).

It is insufficient that the art disclosed components of Applicants' invention. A teaching, suggestion, or incentive must exist to make the combination made by Applicants. *Interconnect Planning Corp. v. Feil*, 774 F.2d 1132, 1143, 227 USPQ 543, 551 (Fed. Cir. 1988).

In view of the foregoing remarks and amendments, the rejection of Claims 1-14 and 31-32 under U.S.C. §103(a) as being unpatentable over *Advanced Compiler Design & Implementation* and *Tip et al.,* have been overcome. Notice to that effect is requested.

The Patent Office rejected Claims 15-24, 33, and 34 under 35 U.S.C. §103(a) as being unpatentable over Fast Static Analysis of C++ Virtual Function Calls, David F. *Bacon et al.*, ACM, 1996, pages 324-341 view of *Tip et al.* The Patent Office states that as to Claim 15, *Bacon et al.* anticipates a method for analyzing a program, comprising: determining an object type that may exist at an execution point of the program, wherein this enables determination of a possible virtual function that may be called. (*Bacon*, page 324 — Introduction and Overview and page 329 Tables results of analysis). The Patent Office states that *Bacon et al.* does not explicitly teach evaluating all possible object types that are created at every instruction of a program and carrying the object types through a stack evaluation. However, the Patent Office states that *Tip et al.* teaches the ability to evaluate all possible object types and the reachability of local methods so that unreachable methods can be removed (i.e. col. 1 lines 40-53). The Patent Office states it would have been obvious for one having ordinary skill in the art to modify *Bacon*'s disclosed system to incorporate the teachings of *Tip*.

The Patent Office states that Claim 20, a computer-readable medium storing computer-executable process steps of a process for analyzing a program, comprising: determining an object type that may exist at an execution point of the program, wherein this enables determination of possible virtual functions that may be called is also taught.

Applicant has amended Claims 15 and 20 in response to the above identified rejection. More specifically, the prior art has tried to address the problem of eliminating, but not all unused virtual functions. *Fast Static* discloses partially eliminating non-virtual functions, but does not eliminate all of these non-virtual function which maintains the problem of having dead code that still remains.

Fast Static teaches the ability to improve C++ programs resolving virtual function calls, thereby reducing compiled code size and reducing program complexity so as to improve human and automated program understanding and analysis. The *Fast Static* study found that using their system, 71 percent of the virtual function calls were resolved. However, almost 30 percent of the dead code still remained and the system still was hampered by unresolved virtual function calls which obviously slow processing speed and take up valuable memory.

Tip et al. teaches a method and system for a program storage device which readably by a machine, gives instructions to perform steps in the creation and construction of a call graph whereby for each method, a set of types of objects that may occur in each method step is determined for each field that may be stored in field. The method determines the allocation sites inside the body of method M and then determines the set of directly called methods M inside the body of method M. Finally, the method determines the set of virtually called methods M' inside the body of method M.

Amended Claim 15 teaches a method for analyzing a program, comprising: determining an object type that may exist at an execution point of the program and evaluating all possible object types that are created at every instruction of a program and carrying the object types through a stack evaluation, wherein this enables determination of a possible virtual function that may be called and allows determination and solutions to program bottlenecks while minimizing virtual functions and dead codes.

Amended Claim 20 teaches a computer-readable medium storing computer-executable process steps of a process for analyzing a program, comprising: determining an object type that may exist at an execution point of the program enabling determination of possible virtual function that may be called and evaluating all possible object types that are created at every instruction of a program and carrying the object types through a stack evaluation, and providing visibility to functions which are required and also to the chain of dependencies.

Neither *Fast Static* nor *Tip et al.* taken singly, or in combination teach or suggest determining an object type that may exist at an execution point of the program and evaluating all possible object types that are created at every instruction of a program and carrying the object types through a stack evaluation, wherein this enables determination of a possible virtual function that may be called as required by Claim 15 and 20. On the contrary, *Fast Static*, as enumerated above does not teach determining an object type that may exist at an execution point and further wherein determination of every possible virtual function is contemplated by that system. Moreover *Tip et al.* does not teach carrying object types through a stack evaluation wherein the process allows for determination of possible virtual functions that may be called and providing visibility to functions which are required and also the chain of dependencies as required by Claim 20. Further, neither *Fast Static* nor *Tip et al.* teach or suggest allowing determination and solutions to program bottlenecks while minimizing virtual functions and dead codes as required by Claim 15.

If is further submitted that the question under §103 is whether the totality of the art would collectively suggest the claimed invention to one of ordinary skill in this art. *In re Simon*, 461 F.2d 1387, 174 USPQ 114 (CCPA 1972).

The elements, even distinguishing elements, are disclosed in the art is alone insufficient. It is common to find elements somewhere in the art. Moreover, most if not all elements perform their ordained and expected functions. The test is whether the invention as a whole, in light of the teaching of the reference, would have been obvious to one of ordinary skill in the art at the time the invention was made. *Connell v. Sears, Roebuck & Co.*, 722 F.2d 1542, 220 USPQ 193 (Fed. Cir. 1983).

It is insufficient that the art disclosed components of Applicant's invention. A teaching, suggestion, or incentive must exist to make the combination made by Applicants. *Interconnect Planning Corp. v. Feil*, 774 F.2d 1132, 1143, 227 USPQ 543, 551 (Fed. Cir. 1988).

In view of the foregoing remarks and amendments, the rejection of Claims 15-24, 33, and 34 under 35 U.S.C. §103(a) obvious over *Fast Static* in view of *Tip et al.* has been overcome and should be withdrawn.

Claims 2-7 and 31 depend from Claim 1; Claims 9-14 and 32 depend from Claim 8; Claims 16-19 and 33 depend from Claim 15; and Claims 21-24 and 34 depend from Claim 20. These claims are further believed to be allowable for the same reasons set forth with respect to independent Claim 1 since each sets forth additional novel elements and steps of Applicant's Method and System from Program Transformation Using Flow Sensitive Type Constraint Analysis.

A Request for Extension of Time for one month extension was filed on April 7, 2009 with the appropriate filing fees. The Commissioner is hereby authorized to deduct any fees due in connection with this response from Deposit Account No. 502191.

In view of the foregoing remarks, Applicant respectfully submits all of the claims in the application are in allowable form and that the application is now in condition for allowance. If any outstanding issues remain, Applicant urges the Patent Office to telephone Applicant's attorney so that the same may be resolved and the application expedited to issue. Applicant requests the Patent Office to indicate all claims as allowable and to pass the application to issue.

Respectfully submitted,
RUTAN & TUCKER

Date: ~~April~~ May 7, 2009

By [illegible]
Hani Z. Sayed
Registration No. 52,544

Rutan & Tucker, LLP
611 Anton Blvd., 14th Floor
Costa Mesa, CA 92626-1931
Telephone (714) 641-5100
Fax (714) 546-9035